

How to Make Custom REST API (Beginner's Guide)

BY HENNER SETYONO / ON 16 JUL 2020

#GENERAL



REST API is a special set of URLs that we can call to get or save data. These URLs are commonly called endpoint.

Every WordPress site comes with some **default API endpoints** like this one for getting recent posts:

```
https://wptips.dev/wp-json/wp/v2/posts
```

Try copy-pasting that to your browser and you will see a bunch of data in **JSON format**. This format can easily be converted into an Array in any programming language.

In this tutorial we will learn how to make a custom endpoint.

Register GET Route

GET Request means **receiving data** from the server.

In this example, we want 2 new endpoints for our Project post-type:

Note: If you would like to follow along with this tutorial, Paste in these code in `functions.php` to generate Project post-type and create some initial data.

1. Static Route at `/projects` to get recent projects:

functions.php

```
add_action( 'rest_api_init', function() {
    register_rest_route( 'my/v1', '/projects', [
        'methods' => 'GET',
        'callback' => 'get_projects',
        'permission_callback' => '__return_true',
    ] );
} );

// Get all projects and assign thumbnail
function get_projects( $params ) {
    $projects = get_posts( [
        'post_type' => 'project',
        'posts_per_page' => 10
    ] );

    foreach( $projects as &$p ) {
        $p->thumbnail = get_the_post_thumbnail_url( $p->ID );
    }

    return $projects;
}
```

PHP

The snippet above will create an endpoint at <https://yoursite.com/wp-json/my/v1/projects>.

You might wonder what is `my/v1`. That is the **namespace** and **version**.

The namespace is to identify a group. You can use anything, but try to keep it short.

The version is to differentiate an updated route. When someday you want to update this API, you should keep the `v1` as is and create a new route with `v2` instead. This is to prevent existing apps that are using v1 from breaking down.

2. Dynamic Route at `/project/[id]` to get a specific project:

functions.php

```
add_action( 'rest_api_init', function() {
    register_rest_route( 'my/v1', '/project/(?P<id>\d+)', [
        'methods' => 'GET',
        'callback' => 'get_project',
        'permission_callback' => '__return_true',
    ] );
} );

// Get single project
function get_project( $params ) {
    $project = get_post( $params['id'] );
    $project->thumbnail = get_the_post_thumbnail_url( $project->ID );
}
```

```
    return $project;
}
```

PHP

The breakdown of `(?P<id>\d+)`:

- `?P<id>` means it will save the value as 'id'.
- `\d+` is the regex validation and means it only accepts numbers. You can read more about regex [in MDN article here](#).

Register POST Route

POST Request means **sending data** to the server.

Continuing from the example above, we need a **Search endpoint** where we can filter by submitting title and/or category.

Here's how:

```
add_action( 'rest_api_init', function() {
    register_rest_route( 'my/v1', '/projects_search', [
        'methods' => 'POST',
        'callback' => 'post_projects_search',
        'permission_callback' => '__return_true',
    ] );
} );

// Search projects
function post_projects_search( $request ) {
    // Get sent data and set default value
    $params = wp_parse_args( $request->get_params(), [
        'title' => '',
        'category' => null
    ] );

    $args = [
        'post_type' => 'project',
        's' => $params['title'],
    ];

    if( $params['category'] ) {
        $args['tax_query'] = [[
            'taxonomy' => 'project_category',
            'field' => 'id',
            'terms' => $params['category']
        ]];
    }
}
```

```
    return get_posts( $args );  
}
```

PHP

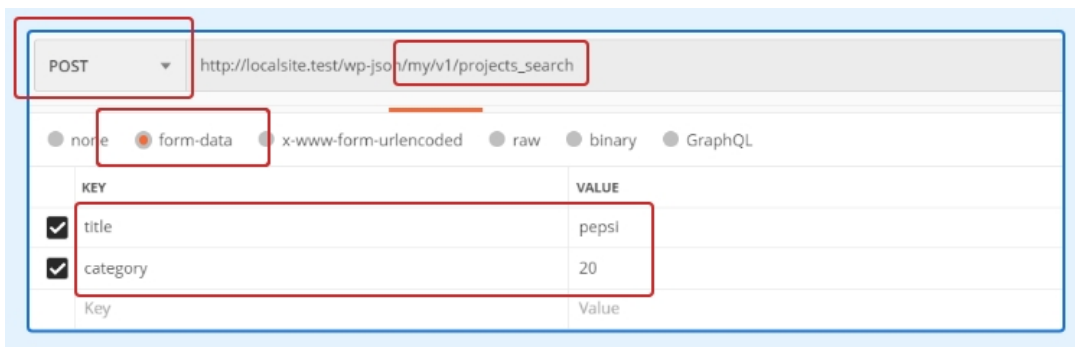
It's difficult to use your browser to simulate POST requests. So I suggest using a software called Postman.

Debugging with Postman

Postman is an amazing free software to try out API requests. First, download and register for an account:

Download Postman

After that, create a new POST request and enter the appropriate data. The screenshot below is the Postman setting for our Search endpoint:



Postman config for POST request

Click SEND and you will see the returned data in the panel below:



The response from POST request above.

If there's an error, move to "Preview" tab to see the error message in a clearer way.

Create a Wrapper Class (Optional)

If the endpoints are related, I recommend putting them in a class. Not only it's tidier it also relieves you from worrying about duplicate method names.

Here's what it looks like when grouping the 3 endpoints above:

functions.php

```
if( !class_exists( 'MyAPI' ) ) {

class MyAPI {
    function __construct() {
        add_action( 'rest_api_init', [ $this, 'init' ] );
    }

    function init() {
        register_rest_route( 'my/v1', '/projects', [
            'methods' => 'GET',
            'callback' => [ $this, 'get_projects' ],
        ] );

        register_rest_route( 'my/v1', '/project/(?P<id>\d+)', [
            'methods' => 'GET',
            'callback' => [ $this, 'get_project' ],
        ] );

        register_rest_route( 'my/v1', '/projects_search', [
            'methods' => 'POST',
            'callback' => [ $this, 'post_projects_search' ]
        ] );
    }

    // Get recent projects
    function get_projects( $params ) {
        $projects = get_posts( [
            'post_type' => 'project',
            'posts_per_page' => 10
        ] );

        foreach( $projects as $p ) {
            $p->thumbnail = get_the_post_thumbnail_url( $p->ID );
        }

        return $projects;
    }

    // Get single project
    function get_project( $params ) {
        $project = get_post( $params['id'] );
        $project->thumbnail = get_the_post_thumbnail_url( $project->ID );
        return $project;
    }
}
```

```

// Search projects
function post_projects_search( $request ) {
    // Get sent data and set default value
    $params = wp_parse_args( $request->get_params(), [
        'title' => '',
        'category' => null
    ] );

    $args = [
        'post_type' => 'project',
        's' => $params['title'],
    ];

    if( $params['category'] ) {
        $args['tax_query'] = [[
            'taxonomy' => 'project_category',
            'field' => 'id',
            'terms' => $params['category']
        ]];
    }

    return get_posts( $args );
}

new MyAPI();
}

```

PHP

. . .

Conclusion

WordPress REST API can be very powerful. It has been used by many plugins like WooCommerce to create an interactive experience.

The Gutenberg editor also used the REST API in many areas such as updating your post without a page refresh.

In my next few posts, I will cover about using JavaScript to call these APIs. This will lead to **Headless WordPress**, a topic most requested by our readers 😊

Useful links:

- List of default API endpoints – [wordpress.org](https://developer.wordpress.org/rest-api/)
- Postman Download – [postman.com](https://www.postman.com/)
- The official tutorial for REST API – [wordpress.org](https://developer.wordpress.org/rest-api/)

api

php

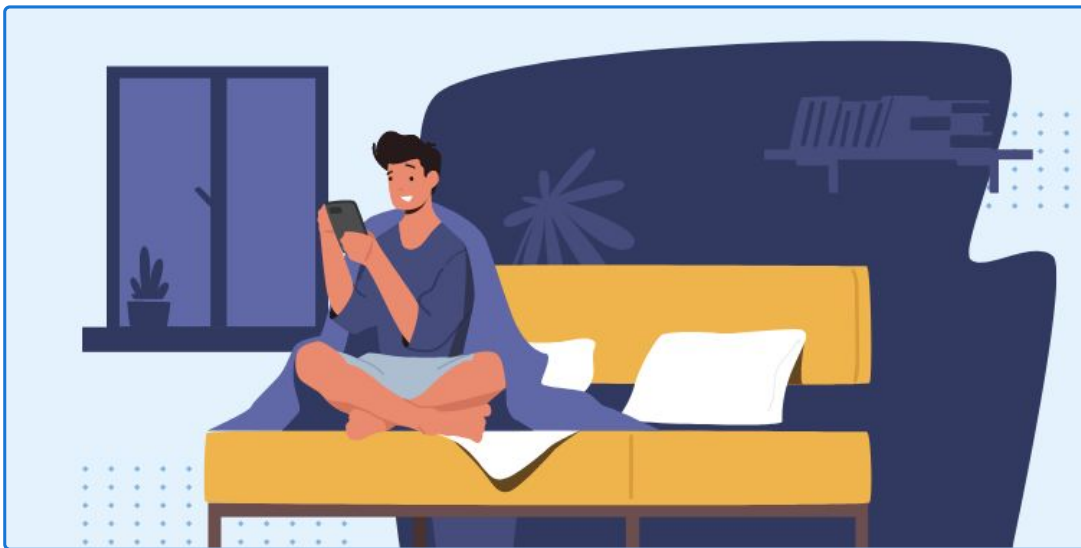


Written by

Henner Setyono

A web developer who mainly build custom theme for WordPress since 2013. Young enough to never had to deal with using Table as layout.

Related Posts



Quick and Easy Dark Mode without Plugin

On 2 Aug 2022



Our WordPress Workflow with Github Action

On 28 Feb 2022 / 2 Comments



What is WordPress Nonce? (Preventing CSRF Attack)







On 14 Aug 2020

Leave a Reply

Name *

Email *

Website

B **I**      Block 

- ☐ Save my name, email, and website in this browser for the next time I comment.
- ☐ Notify me of follow-up comments by email.

Post Comment



Design by [Pixel Studio](#)

About wpTips.dev

wpTips is a non-profit blog focused on **advanced** WordPress tutorial in a concise and beautiful way.

We want to show the world the full potential of our beloved CMS.

Join Our Community

We are a team of enthusiasts who enjoy sharing our knowledge.

Let us know at hello@wptips.dev if you are interested to write some tips!

Copyright © 2022 WPTips.dev